

Задача Cutting. Разрезание графа

Имя входного файла: `cutting.in`
 Имя выходного файла: `cutting.out`
 Ограничение по времени: 2 секунды
 Ограничение по памяти: 64 мегабайта

Дан неориентированный граф. Над ним в заданном порядке производят операции следующих двух типов:

- **cut** — разрезать граф, то есть удалить из него ребро;
- **ask** — проверить, лежат ли две вершины графа в одной компоненте связности.

Известно, что после выполнения всех операций типа **cut** рёбер в графе не осталось. Найдите результат выполнения каждой из операций типа **ask**.

Формат входного файла

Первая строка входного файла содержит три целых числа, разделённые пробелами — количество вершин графа n , количество рёбер m и количество операций k ($1 \leq n \leq 50\,000$, $0 \leq m \leq 100\,000$, $m \leq k \leq 150\,000$).

Следующие m строк задают рёбра графа; i -ая из этих строк содержит два числа u_i и v_i ($1 \leq u_i, v_i \leq n$), разделённые пробелами — номера концов i -го ребра. Вершины нумеруются с единицы; граф не содержит петель и кратных рёбер.

Далее следуют k строк, описывающих операции. Операция типа **cut** задаётся строкой “**cut** u v ” ($1 \leq u, v \leq n$), которая означает, что из графа удаляют ребро между вершинами u и v . Операция типа **ask** задаётся строкой “**ask** u v ” ($1 \leq u, v \leq n$), которая означает, что необходимо узнать, лежат ли в данный момент вершины u и v в одной компоненте связности. Гарантируется, что каждое ребро графа встретится в операциях типа **cut** ровно один раз.

Формат выходного файла

Для каждой операции **ask** во входном файле выведите на отдельной строке слово “**YES**”, если две указанные вершины лежат в одной компоненте связности, и “**NO**” в противном случае. Порядок ответов должен соответствовать порядку операций **ask** во входном файле.

Пример

cutting.in	cutting.out
3 3 7	YES
1 2	YES
2 3	NO
3 1	NO
ask 3 3	
cut 1 2	
ask 1 2	
cut 1 3	
ask 2 1	
cut 2 3	
ask 3 1	

Задача Distance. Расстояние между верши-